Journal of Al-Azhar University Engineering Sector

Vol. 78, No. 21, January 2026, 128 - 150

Original Article

COMPARATIVE ANALYSIS OF MACHINE LEARNING MODELS FOR PREDICTIVE MAINTENANCE: CHALLENGES AND OPPORTUNITIES

Manal Helal*, Stilianos Vidalis, Baari Ali, Frank Foerster, Eric Chiejina

School of Physics, Engineering and Computer Science, University of Hertfordshire, College Lane, Hatfield, UK.

*** Correspondence: m.helal@herts.ac.uk**

Citation:

M. Helal, S. Vidalis, B. Ali, F. Foerster, E. Chiejina, "Comparative Analysis of Machine Learning Models for Predictive Maintenance: Challenges and Opportunities" Journal of Al-Azhar University Engineering Sector, vol. 21 (78), pp. 128-150, 2026.

Received: **04 June 2025**

Revised: **29 August 2025**

Accepted: **14 September 2025**

Doi: 10.21608/auej.2026.482411

ABSTRACT

Predictive maintenance of Hard Disk Drives (HDDs) using machine learning is vital for preventing data loss and operational downtime in data centres. Sequential SMART (Self-Monitoring, Analysis, and Reporting Technology) data provides a rich source for forecasting impending failures. However, a clear consensus on the most effective modelling approach is absent, constituting a knowledge gap caused by studies that focus on a single paradigm without a comprehensive, comparative benchmark across multiple task types. This research aims to systematically evaluate and rank the performance of over 20 models across regression, classification, time-series forecasting, and anomaly detection tasks to identify the optimal strategies for predicting HDD failures. We conducted an extensive empirical study, employing tree-based models, deep learning architectures, including LSTMs (Long Short-Term Memory networks), GRUs (Gated Recurrent Units), and Temporal Fusion Transformers (TFT), as well as various autoencoders. A full hyperparameter sweep was performed for time-series models to ensure robust comparisons. Tree-based models excelled in static analysis, while deep learning was superior for temporal sequences. The TFT model achieved the highest average accuracy (86.12%) and best generalisation. The GRU model attained the highest failure recall (92.52% peak accuracy). Conversely, anomaly detection methods demonstrated only moderate performance (65-73% accuracy). The key contribution of this work is a definitive, evidence-based model selection framework that addresses this gap. The TFT architecture is identified as the most robust and efficient model for analysing temporal SMART data. Practitioners should prioritise tree-based models for static tabular data, TFT for accurate and reliable forecasting, and GRU for maximising failure detection sensitivity, thereby significantly enhancing predictive maintenance frameworks..

KEYWORDS: Predictive Maintenance; GPU Acceleration; Regression; Classification; Time Series; Auto Encoders; Sequence-2-Sequence.

تحليل مقارن لنماذج التعلم الآلي للصيانة التنبؤية: التحديات والفرص

منال هلال*، ستيليانوس فيداليس، باري علي، فرانك فورستر، إريك تشيجينا

كلية الفيزياء والهندسة وعلوم الحاسوب، جامعة هيرتفوردشاير، كوليدج لين، هاتفيلد، المملكة المتحدة

* **البريد الالكتروني للباحث الرئيسي:** m.helal@herts.ac.uk

الملخص

تهدف هذه الدراسة إلى تحسين خواص الخرسانة والنظام الإنشائي لأغطية غرف تفتيش الاتصالات وذلك لزيادة مقاومة الخرسانة ومتانتها وزيادة مقاومة يُعدّ الصيانة التنبؤية لمحركات الأقراص الصلبة باستخدام تقنيات التعلّم الآلي أمرًا بالغ الأهمية لمنع فقدان البيانات وتوقف العمليات في مراكز البيانات. توفر بيانات تقنية SMART (تقنية المراقبة الذاتية والتحليل والإبلاغ) المتسلسلة مصدرًا غنيًا للتنبؤ بالأعطال الوشيكة. مع ذلك، لا يوجد إجماع واضح على أنجع أساليب النمذجة، مما يُشكّل فجوة معرفية ناتجة عن دراسات تُركّز على نموذج واحد دون وجود معيار شامل ومقارن عبر أنواع مهام متعددة. يهدف هذا البحث إلى التقييم المنهجي وترتيب أداء أكثر من ٢٠ نموذجًا في مهام الانحدار والتصنيف والتنبؤ بالسلاسل الزمنية واكتشاف الشذوذ، وذلك لتحديد الاستراتيجيات الأمثل للتنبؤ بأعطال محركات الأقراص الصلبة. أجرينا دراسة تجريبية شاملة، باستخدام نماذج قائمة على الأشجار، وهياكل التعلّم العميق، بما في ذلك شبكات الذاكرة طويلة المدى (LSTM) ووحدات التكرار البوابية (GRU) ومحولات الدمج الزمني (TFT)، بالإضافة إلى العديد من المشفرات التلقائية. أُجري مسح شامل للمعلمات الفائقة لنماذج السلاسل الزمنية لضمان مقارنات دقيقة. تفوقت النماذج القائمة على الأشجار في التحليل الثابت، بينما أظهر التعلم العميق أداءً أفضل في تحليل التسلسلات الزمنية. حقق نموذج TFT أعلى متوسط دقة (٨٦,١٢٪) وأفضل قدرة على التعميم. وحقق نموذج GRU أعلى معدل استدعاء للأعطال (٩٢,٥٢٪ دقة قصوى). في المقابل، أظهرت طرق كشف الشذوذ أداءً متوسطًا فقط (دقة تتراوح بين ٦٥٪ و٧٣٪). تتمثل المساهمة الرئيسية لهذا العمل في إطار عمل نهائي لاختيار النموذج قائم على الأدلة، والذي يسد هذه الفجوة. تم تحديد بنية TFT كأكثر النماذج قوة وكفاءة لتحليل بيانات SMART الزمنية. ينبغي على الممارسين إعطاء الأولوية للنماذج القائمة على الأشجار للبيانات الجدولية الثابتة، وTFT للتنبؤ الدقيق والموثوق، وGRU لزيادة حساسية كشف الأعطال إلى أقصى حد، مما يُحسّن بشكل كبير أطر الصيانة التنبؤية.

الكلمات المفتاحية : الصيانة التنبؤية؛ تسريع وحدة معالجة الرسومات؛ الانحدار؛ التصنيف؛ السلاسل الزمنية؛ المشفرات التلقائية؛ التسلسل إلى التسلسل.

1. INTRODUCTION

Time series datasets are prevalent in various application domains, including, but not limited to, sales or continuous variables forecasting, Internet of Things (IoT) sensor networks for surveillance and remote sensing for early intervention and prevention, health monitoring using wearable health devices, and notably in predictive maintenance (PdM). PdM has emerged as a pivotal strategy in modern industrial operations, driven by the increasing complexity and criticality of machinery and equipment across various sectors. Traditional maintenance approaches, such as reactive or scheduled maintenance, often result in suboptimal performance, higher operational costs, and increased downtime. Reactive maintenance, also known as Run-to-Failure (R2F), addresses failures only after they occur, resulting in costly unplanned outages and significant production losses. Scheduled maintenance, also labelled Preventive Maintenance (PM), while proactive, can result in unnecessary downtime and the cost of maintenance work if performed too frequently or without considering the actual condition of the equipment. Others also identify Condition-Based Maintenance (CBM) Systems that are maintained based on simple rules using equipment information [1].

In contrast, Predictive Maintenance (PdM) leverages advanced technologies such as the Internet of Things (IoT), data analytics, data mining, Artificial Intelligence (AI), machine learning (ML) and Deep Learning (DL) to predict failures or deteriorations before they happen, optimising maintenance schedules and minimising unexpected breakdowns. Predictive maintenance models can accurately estimate a machine's Remaining Useful Life (RUL) and predict potential failures by analysing historical and real-time data from various sensors. This capability enhances operational efficiency, significantly reduces maintenance costs, and prevents catastrophic failures that could result in extensive damage and costly repairs. Identifying historical diagnostic information from historical data can result in identifying the failing profiles for further prevention, such as reducing operational temperatures or others [2].

The financial implications of predictive maintenance are substantial. According to industry reports, companies implementing predictive maintenance can increase uptime by up to 20%, enhance productivity by 25%, decrease breakdowns by 70%, and reduce maintenance expenses by 25% [3]. However, the adoption of predictive maintenance is not without its challenges. These include the initial implementation costs, such as the considerable investment in data acquisition and model development, the complexity of the systems, the required computational resources, the time for setup, and the potential for prediction inaccuracies [4]. Despite these hurdles, the substantial cost savings and operational benefits of predictive maintenance over traditional methods often make the investment worthwhile. For instance, in the oil industry, the application of DL-based anomaly detection for Electric Submersible Pumps (ESPs) is critical. Current predictive maintenance techniques are often hampered by high false alarm rates, which can lead to alert fatigue and missed critical failures [5], [6]. These limitations result in significant non-producing time (NPT); industry reports highlight that unplanned downtime can cost a major offshore platform millions of dollars per event and underscore the urgent need for improved AI-driven solutions to mitigate these losses [5]. The adoption of advanced data-driven approaches, particularly ensemble learning and deep learning models, is a key research direction to overcome these challenges. Recent studies demonstrate that such models can significantly enhance prediction accuracy and reliability for oil and gas equipment, directly addressing the high costs of downtime and equipment failure [6], [7].

As industries continue to embrace digital transformation, the development and application of predictive maintenance models have become increasingly sophisticated. The advent of advanced machine learning techniques, coupled with powerful computational resources, has enhanced the accuracy and efficiency of predictive maintenance systems. Unlike deep learning models, traditional machine learning algorithms have more straightforward computational requirements and are easier to interpret as they do not involve deep architectures and complex optimisation techniques. However, they do not perform as well as deep learning models for larger and higher-dimensional data. This paper explores the effectiveness of various machine learning models and hardware configurations for predictive maintenance, providing valuable insights into their performance and computational requirements. By systematically evaluating these models, we aim to contribute to the ongoing efforts to refine predictive maintenance practices and help organisations achieve excellent operational reliability and cost efficiency.

The subsequent sections of this paper are structured as follows: Section 2 provides a literature review on predictive maintenance, covering traditional statistical approaches, machine learning methods, and deep learning architectures. Section 3 details the methodology, including the dataset description, data preprocessing steps, and the experimental setup for each of the four machine learning tasks. Section 4 presents and critically discusses the experimental results, analysing the performance of various models across different tasks. Finally, Section 5 concludes the paper by summarising the key findings, highlighting the practical implications, and outlining future research directions.

2. LITERATURE REVIEW

Predictive maintenance (PdM) has evolved significantly with advancements in machine learning (ML) and deep learning (DL), enabling more accurate failure predictions and optimised maintenance schedules. This section reviews key methodologies, including traditional statistical approaches, traditional ML models, and modern DL architectures, highlighting their applications and limitations in PdM.

2.1. Traditional Statistical Approaches

Early approaches to time series analysis relied on statistical methods such as autoregressive (AR) models [8] and exponential smoothing [9], [10]. These methods are computationally efficient and interpretable but often struggle with non-linear patterns and high-dimensional data. Structural time-series models, which decompose data into trend, seasonality, and noise components, have also been widely used [11]. Gaussian processes [12] represent a classical machine learning approach with strong statistical foundations, offering robust probabilistic frameworks for time-series prediction. Recent extensions, such as deep Gaussian processes [13], bridge traditional statistical methods with modern deep learning techniques.

2.2. Traditional Machine Learning Methods

Traditional ML algorithms, such as kernel regression [13] and Support Vector Regression (SVR) [14], have been applied to time-series forecasting. These methods excel in capturing non-linear relationships but require careful feature engineering. Decision trees and ensemble methods like Random Forests [15] and XGBoost [16] are robust for handling heterogeneous data and feature interactions, making them popular for PdM tasks [17]. However, their performance often plateaus with high-dimensional or sequential data, where DL models typically outperform them.

2.3. Deep Learning for Time Series and Predictive Maintenance

Deep learning models, particularly recurrent neural networks (RNNs), have revolutionised sequential data analysis. Long Short-Term Memory (LSTM) networks [18], [19] and Gated Recurrent Units (GRUs) [20] address the vanishing gradient problem in traditional RNNs, enabling longer sequence modelling. Bidirectional LSTMs (BiLSTMs) [21] further enhance performance by capturing both past and future contexts. For spatial-temporal data, 1D convolutional neural networks (CNNs) [22] combined with LSTMs (CNN-LSTM) [23] extract local patterns and temporal dependencies effectively. Transformers [24] have emerged as a powerful alternative, leveraging self-attention mechanisms to model long-range dependencies without recurrence. Their applications in time-series forecasting [25] and PdM [26] demonstrate superior performance in capturing complex patterns. WaveNet [27], originally designed for audio generation, has been adapted for time-series prediction due to its dilated convolutions, which efficiently model multi-scale dependencies.

2.4. Anomaly Detection and Autoencoders

Autoencoders (AEs) are widely used for anomaly detection in PdM. Variational autoencoders (VAEs) [28] introduce probabilistic latent spaces, while sparse autoencoders [29] enforce sparsity constraints for robust feature learning. Attention-based AEs [30] and transformer AEs [24] further improve reconstruction fidelity by focusing on salient features. Comparative studies [31] highlight the trade-offs between model complexity and detection accuracy in industrial applications.

2.5. Challenges and Gaps

Despite these advancements, challenges remain:

1. **Imbalanced Data**: PdM datasets often exhibit severe class imbalance, necessitating techniques like synthetic minority oversampling (SMOTE) [32] or cost-sensitive learning [33].
2. **Interpretability**: DL models, while accurate, are often seen as "black boxes." Hybrid approaches combining DL with explainable AI (e.g., LIME [34]) are gaining traction.
3. **Computational Costs**: Training complex models like transformers requires significant resources, motivating research into lightweight architectures enabling better accuracy-efficiency trade-off (e.g., MobileNet for edge devices; [35]).
4. **IoT Big Data Vs and failure modelling**: The Vs (Volume, Velocity, Variety, Veracity, and Value) require careful handling from one industry to another. The sequence length appropriate for failure buildup will vary depending on the machines being monitored. The handling of

missing values and noisy data will differ, and the computational cost for the different models will also vary.

2.6. Recent Trends

Recent work explores federated learning for PdM [36] to address data privacy concerns and reinforcement learning (RL) for adaptive maintenance scheduling [37]. The integration of physics-informed ML [38] is also promising, combining domain knowledge with data-driven models. Furthermore, the increasing adoption of digital twins [39] and the Internet of Things (IoT) [48] in industrial settings provides unprecedented opportunities for real-time data collection and analysis, which are crucial for advanced predictive maintenance. The development of robust and scalable data pipelines [49] is crucial for handling the volume and velocity of data generated by these systems. Additionally, the application of transfer learning [40], meta-learning and multi-modal approaches [41] is gaining traction, allowing models to leverage knowledge from different domains or tasks, thereby reducing the need for extensive historical data in new deployment scenarios. Adversarial learning [42] is also being explored to enhance model robustness against noisy or adversarial sensor data, ensuring reliability even under suboptimal conditions.

3. METHODOLOGY

Our study utilised a subset of the publicly available Backblaze Basic Drive Information dataset, a rich source of daily S.M.A.R.T. (Self-Monitoring, Analysis, and Reporting Technology) attribute readings from a large population of operational hard drives. Each record corresponds to a single disk on a specific day, identified by model, serial number, and capacity. The target variable is a binary failure label, where '1' signifies the final day preceding a drive failure and '0' indicates normal operation.

Table 1: Backblaze Dataset Summary

Description	Value	Notes
Total Raw Records	~3.2 million	From 2017 data
Number of Features	95, 28 after preprocessing	S.M.A.R.T. attributes
Normal Instances (Class 0)	3,196,318	
Failure Instances (Class 1)	234	Highly Imbalanced
Training Period	Q1-Q3 2017	
Testing Period	Q4 2016	

Data preprocessing was critical for ensuring data quality and experimental consistency. Incomplete rows containing missing values were removed to preserve data integrity. The profound class imbalance (**Table 1**) posed a significant challenge for classification tasks. To mitigate this, we down sampled the majority 'normal' class to create a balanced training set, a necessary step to prevent models from simply learning always to predict the majority class. For regression tasks, we focused exclusively on disks that eventually failed, calculating the Remaining Useful Life (RUL) for each daily reading as the number of days remaining until the failure event.

For time-series and autoencoder tasks, we constructed sequences of consecutive daily sensor readings. Gaps in the sequential data for a given disk were filled using a forward-fill method, propagating the last observed value forward. Sequences that ended in a failure event were labelled '1', while sequences from consistently healthy drives were labelled '0'.

3.1. Experimental Framework Overview

To conduct a comprehensive evaluation, we designed four distinct machine learning tasks that represent common predictive maintenance scenarios. The models evaluated for each task are summarised in **Error! Reference source not found.**, providing a clear map of our experimental design. A detailed explanation of these models is presented in Supplement 1.

Table 2: Summary of Experimental Tasks and Models

Task	Objective	Models Evaluated
1. Regression	Predict Remaining Useful Life (RUL)	SVR, Linear, Ridge, Lasso, Elastic Net, Decision Tree, Random Forest, XGBoost, ANN
2. Classification	Binary state (Fail/Normal) classification	Logistic Regression, SVM, Random Forest, XGBoost, LightGBM, CatBoost, ANN
3. Time Series Classification	Predict failure in the future (Fail/Normal in look-ahead days)	LSTM, GRU, BiLSTM, CNN-LSTM, Transformer, WaveNet, TFT
4. Anomaly Detection	Identify anomalies (to predict fail) via reconstruction error	Dense AE, Sparse AE, VAE, LSTM AE, CNN AE, Attention AE, Transformer AE

3.2 Model Architectures and Training Configuration

3.2.1 Regression and Classification Models

For the traditional tasks of regression and classification, we employed a suite of established algorithms implemented in scikit-learn [38] and XGBoost [44]. This selection was designed to benchmark a spectrum of approaches, from simple linear baselines to complex, state-of-the-art ensembles.

- **Linear Models (Linear Regression, Ridge, Lasso, Elastic Net):** These were included as fundamental baselines. Ridge and Lasso regression introduce L2 and L1 regularisation, respectively, to prevent overfitting in the high-dimensional S.M.A.R.T. data. Elastic Net was selected to combine the benefits of both L1 and L2 regularisation, which is often robust for correlated predictive features commonly found in sensor data.
- **Support Vector Machines (SVR, SVM):** Chosen for their effectiveness in high-dimensional spaces and ability to model non-linear relationships through kernel functions. They serve as a strong non-linear baseline against the tree-based methods.
- **Tree-Based Ensembles (Random Forest, XGBoost, LightGBM, CatBoost):** These were the core models of our study due to their consistent top performance on tabular data problems. **Random Forest** [13] was selected for its robustness against overfitting and ability to capture complex interactions. **XGBoost** [14] was chosen as a leading gradient-boosting implementation known for its speed and performance. **LightGBM** and **CatBoost** were included as modern alternatives that offer efficient handling of large datasets and sophisticated handling of categorical features, respectively, providing a comprehensive comparison of leading ensemble techniques.
- **Artificial Neural Networks (ANN):** A simple multi-layer perceptron was implemented as a baseline deep learning model. Its performance provides a reference point for determining whether the complexity of deep learning is justified for these specific tabular data tasks or if more interpretable ensemble methods suffice.

Hyperparameter tuning for these models was conducted using GridSearchCV and RandomizedSearchCV from scikit-learn, optimising key parameters such as the number of estimators, maximum depth, learning rate (for boosting algorithms), and regularisation strengths.

3.2.2 Deep Learning Models for Time Series Classification

For predicting future failures based on sequences of past readings, we implemented several advanced deep learning architectures in Keras/TensorFlow as detailed in **Table 3**. This selection represents the state-of-the-art in sequence modelling, from recurrent networks to attention-based models.

- **Long Short-Term Memory (LSTM):** The canonical choice for sequence modelling, selected as a strong baseline for capturing temporal dependencies and long-range patterns in the degradation signals [43].
- **Gated Recurrent Unit (GRU):** Included as a more computationally efficient alternative to the LSTM, with a simplified gating mechanism, to test if comparable performance could be achieved with lower resource costs [20].
- **Bidirectional LSTM (BiLSTM):** Chosen to enhance the LSTM's capabilities by processing sequences both forwards and backwards, allowing the model to leverage contextual information from both past and future points within the sequence for each prediction [21], [44].
- **CNN-LSTM Hybrid:** This architecture was specifically selected to exploit the strengths of both convolutional and recurrent layers. The initial 1D CNN layers efficiently extract local patterns and features within short time windows, which are then fed into the LSTM layers to model the longer-term temporal dependencies between these extracted features. We hypothesised this would be particularly effective for the localised spike patterns often present in degradation data [45].
- **Transformer:** Implemented to test the efficacy of self-attention mechanisms for this task. Transformers can weigh the importance of all time steps in the sequence simultaneously, potentially capturing complex, long-range dependencies more effectively than recurrent networks, albeit often at a higher computational cost [24].
- **WaveNet:** Chosen for its unique use of dilated causal convolutions, which allow for an exponentially large receptive field to capture very long-range dependencies with relatively few layers. This makes it highly efficient and theoretically well-suited for modelling the long, gradual degradation processes that lead to failure [46], [47].
- **Temporal Fusion Transformer (TFT):** Implemented to explore a modern architecture designed explicitly for interpretable, multi-horizon forecasting. Its combination of variable selection, sequence-to-sequence processing, and interpretable attention mechanisms offers a potential advantage in both performance and providing insights into the prediction [48].

All models were trained on sequences of sequence_length days (ranging from 5 to 14) to predict a failure look_ahead days (1 to 5) into the future. Key architectural details are summarised in **Table 3.** The time complexity of the models applied is expressed in terms of these variables: T: Sequence length (number of time steps), N: Batch size (number of samples processed simultaneously), M: Number of units in a layer (hidden dimension), C: Number of input channels/features, and K: Kernel size in convolutional layers.

Table 3: Key Architecture Details for Time Series Models

Model	Key Architecture Details	Parameters	Theoretical Time Complexity Per Layer	Average Time per Epoch (s)
LSTM	2 LSTM layers (68, 68 units), 20% Dropout, 2 Dense layers (34, 1)	253.68 KB	$O(T \times N \times M^2)$	2.07
GRU	2 stacked GRU layers (84, 84 units), 20% Dropout, 2 Dense layers (42, 1)	289.74 KB	$O(T \times N \times M^2)$	2.07
BiLSTM	2 stacked Bidirectional LSTM layers (48 fwd/bwd), 20% Dropout, 2 Dense layers (48, 1)	345.38 KB	$O(T \times N \times M^2)$	3.09
CNN-LSTM	Conv1D(48) + MaxPooling, Dropout, LSTM(60), 2 Dense layers (30, 1)	123.14 KB	$O(T \times N \times (K \times C \times M + M^2))$	2.05
Transformer	Dense(64), MultiHeadAttention, Add+Norm, FFN(32->64), Add+Norm, GAP, Dense(1)	88.88 KB	$O(T^2 \times N \times M)$	1.30
WaveNet	5 residual blocks with dilated causal convolutions (filters=48), skip connections, GAP	178.50 KB	$O(T \times N \times M)$	1.34
TFT	Variable Selection (per-feature Dense), Multi-Head Attention, LSTM(24), Simplified GRN	35.44 KB	$O(T^2 \times N \times M)$	5.99

Sequence Processing: Input sequences were constructed using a consecutive sequence length of days (ranging from 5 to 16 days) to predict a failure in the next look-ahead days (1 to 5 days) into the future.

3.2.3. Autoencoder Models for Anomaly Detection

Six autoencoder variants were implemented for unsupervised anomaly detection, where a high reconstruction error identifies anomalies as detailed in **Table 4**. The time complexity variables are: T, the sequence length; N, the batch size; M, the hidden dimension; K, the kernel size; and C, the number of channels. The reconstruction error threshold was used in a supervised manner by sorting the MSE of the reconstructed sequences in the test set and identifying as failed disks the sequences with the highest MSE after the number of regular disks in the test set. This enabled the AE models to be used for classifying sensor data as usual or failing, based on the highest anomaly in reconstruction compared to the regular disks.

Table 4: Key Architecture Details for Autoencoder Models

Model	Encoder	Decoder	Loss	Parameters	Theoretical Time Complexity Per Layer	Average Time per Epoch (s)	Rationale
Dense	FC (21→64→32)	FC (32→64→21)	MSE	27.21 KB	O(N × M)	0.39	Simple baseline
Sparse	FC (21→64) + L1 reg.	FC (64→21)	MSE	10.83 KB	$O(N \times M)$	0.26	Learning compact features
VAE	FC (128,64) → (μ, logσ²)	FC (64,128)	MSE + KL	427.06 KB	$O(N \times M)$	0.33	Probabilistic latent space
LSTM	LSTM(128)	RepeatVector + LSTM(128)+LSTM(21)	MAE	863.22 KB	$O(T \times N \times M^2)$	1.75	Temporal sequences
CNN1D	Conv1D(160)+Pool+Conv1D(80)+Pool+Conv1D(4)	UpSample+Conv1D(80)+UpSample+Conv1D(160)+Conv1D(21)	MSE	708.85 KB	$O(T \times N \times K \times C)$	2.66	Local spatial patterns
Attention	FC (21→64) + Self-Attention	FC (64→21)	MSE	10.83 KB	$O(T^2 \times N \times M)$	0.37	Attention mechanisms
Transformer	MultiHeadAttention + FC(32) + LayerNorm (3 blocks)	Flatten + FC(16) + FC(336) + Reshape + MultiHeadAttention + FC(32)	MSE	242.65 KB	$O(T^2 \times N \times M)$	3.11	Full transformer architecture

3.3. Hyperparameter Optimisation and Evaluation Metrics

Hyperparameter optimisation was a critical step. We employed GridSearchCV for an exhaustive search on smaller parameter spaces and RandomizedSearchCV for efficiency on larger spaces. Bayesian optimisation was also explored for its sample efficiency. All models were evaluated using appropriate metrics: Mean Squared Error (MSE) for regression tasks, and Accuracy, Precision, Recall, and F1-Score for classification and anomaly detection tasks. The F1-score for the minority 'failure' class was particularly emphasised due to the inherent class imbalance.

4. RESULTS AND DISCUSSIONS

The experiments utilised the Backblaze dataset from 2017, covering all three quarters for training, with data from Q4 2016 reserved for testing. The 2017 dataset was further divided into an 80:20 training-to-testing ratio for the regression models. The results indicate that fine-tuning hyperparameters significantly enhances performance compared to trial-and-error approaches.

4.1. Regression and Classification Results

Regression experiments utilising the Backblaze dataset from 2017 (all four quarters for training, with Q4 2016 reserved for testing) demonstrated distinct performance patterns across model architectures, as illustrated in **Fig. 1**. Traditional regression models, including Lasso (MSE: 6576.155) and Elastic Net (MSE: 6585.827), provided solid baseline performance in predicting Remaining Useful Life (RUL). However, ensemble methods significantly outperformed these linear approaches, with Random Forest (MSE: 5705.456) and Decision Trees (MSE: 5753.634) capturing non-linear relationships more effectively. Hyperparameter tuning yielded substantial improvements, as evidenced by the fine-tuned Random Forest achieving an MSE of 35.978 and the optimised Decision Tree reaching 235.817, representing 99.4% and 95.9% error reductions, respectively, from their baseline counterparts.

Neural networks presented both challenges and insights throughout the experimentation process. Initial architectures with 3 and 5 layers produced excessively high errors (4.14×10^{20} and 2.64×10^{20}, respectively), indicating significant optimisation difficulties. Through systematic hyperparameter tuning via RandomizedSearchCV with 3-fold cross-validation, the optimal configuration emerged as a 5-layer architecture with 10 neurons per layer, a learning rate of 1×10^{-4}, a batch size of 32, and 100 training epochs. This best neural network model achieved a final MSE of 3.70×10^{18} on the independent test set. While this represents a substantial improvement over untuned neural networks, the persistent high error suggests either fundamental architectural limitations for this regression task or the need for more sophisticated regularisation techniques. The results collectively demonstrate that while neural networks offer theoretical advantages, tree-based ensembles provided more reliable and interpretable performance for RUL prediction in this application.

Model	MSE
Support Vector Regression	6903.446
Linear Regression	6616.934
Ridge Regression	6615.736
Elastic Net	6585.827
Lasso Regression	6576.155
Decision Tree	5753.634
Random Forest	5705.456
Best Fine Tuned Hyperparameters Decision Tree	235.8174
Best Fine Tuned Hyperparameters Random Forest	35.97758
NN - 3 Layers	4.14E+20
NN - 5 Layers	2.64E+20
Best NN Model	3.7E+18

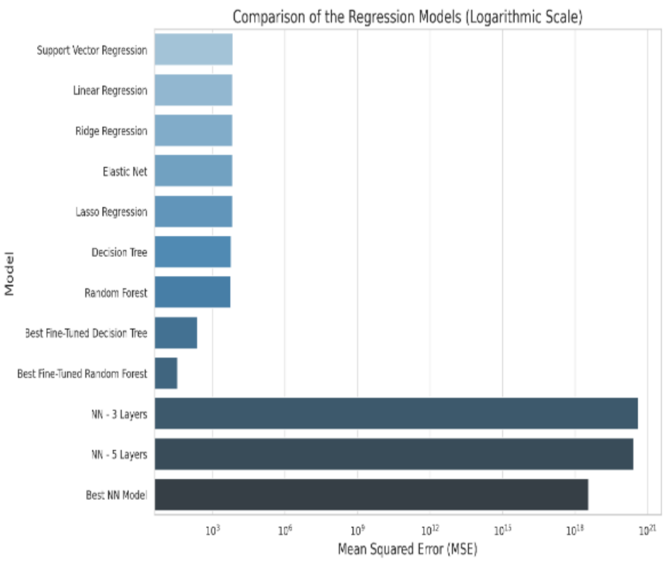


Fig. 1: Regression Models MSE Comparison

In the second set of experiments, classification models were evaluated using one-day SMART readings to predict normal or failing status for the following day, leveraging both CPU and GPU resources on a Google Colab A100 runtime. The best-performing neural network architecture was identified through an extensive hyperparameter optimisation process using RandomizedSearchCV with 3-fold cross-validation across 10 different parameter combinations. The search varied multiple architectural parameters: number of layers (3-6), neurons per layer (10-150), learning rate (0.0001-0.1), batch size (16-64), and training epochs (10-30). The optimal configuration consisted of 3 hidden layers with 150 neurons each, a learning rate of 0.00046, a batch size of 32, and 20 training epochs.

As illustrated in **Fig. 2**, tree-based ensembles demonstrated superior performance, with Random Forest on GPU achieving the highest accuracy of 86.99% and XGBoost reaching an accuracy of 85.63%. The best-performing neural network on GPU achieved an accuracy of 45.44% with a substantial training time of 4.674 seconds, suggesting fundamental architectural limitations in capturing the relevant patterns from SMART data despite extensive hyperparameter tuning. This performance gap highlights the challenge of applying standard feedforward networks to this specific predictive maintenance application.

The class balancing resulted in an accuracy improvement from 100% to approximately 86%, and the F1-score of the "fail" class improved from 5% to 82% using the XGBoost model, demonstrating the model's most significant sensitivity to class imbalance. The down-sampling strategy utilised 281 normal instances and 234 failure samples, creating a balanced dataset that prevented the artificial inflation of accuracy metrics while maintaining representative characteristics of the original data distribution. This approach proved particularly effective for the gradient boosting methods, which achieved consistent performance in the 86-87% accuracy range after balancing. The confusion matrices of the XGBoost model before and after balancing are shown in **Table 5**. The model's performance comparison is illustrated in **Fig. 3**. The detailed experimental results of these models are presented in Supplement 2.

Model	Accuracy	Training Time (sec)
XGBoost/ GPU	0.86	0.104907
Logistic Regression/ GPU	0.55	0.024041
Logistic Regression/ CPU	0.67	0.060373
Random Forest/ GPU	0.87	0.481323
Random Forest/ CPU	0.86	0.351029
SVM/ GPU	0.54	0.818962
SVM/ CPU	0.54	0.321264
LightGBM/ CPU	0.86	0.091085
CatBoost/ GPU	0.87	1.967428
CatBoost/ CPU	0.86	0.226136
MLPClassifier/ CPU	0.55	0.543827
Best NN/ GPU	0.45	4.674185

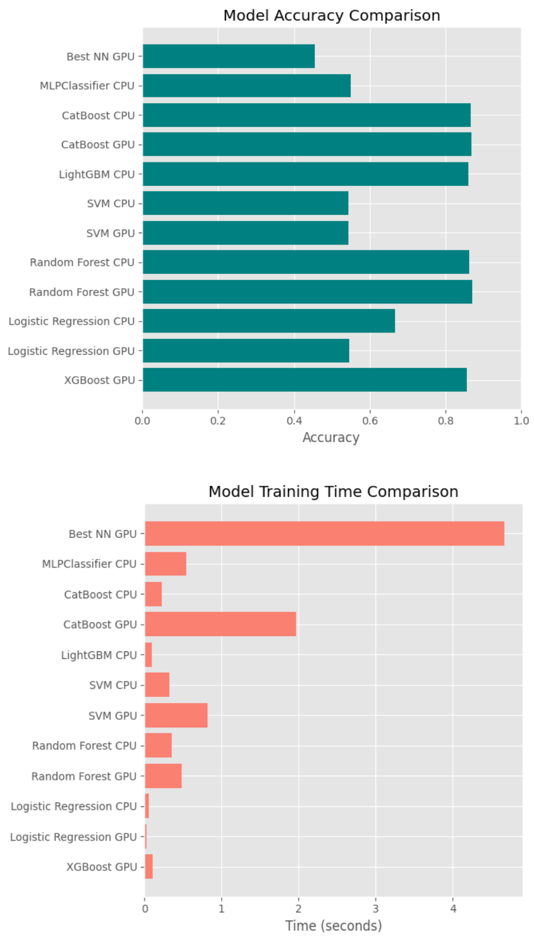


Fig. 2: Comparing Training Time and Accuracy of Classification Models

Table 5: XGBoost Confusion Matrices a) before balancing, and (b) after balancing

(a)					(b)				
Class	**Precision**	**Recall**	**F1-Score**	**Support**	**Class**	**Precision**	**Recall**	**F1-Score**	**Support**
Normal	**1**	**1**	**1**	**3196318**	**Normal**	**0.81**	**0.99**	**0.89**	**281**
Fail	**0.75**	**0.03**	**0.05**	**234**	**Fail**	**0.98**	**0.71**	**0.82**	**234**
	accuracy		**1**	**3196552**		**accuracy**		**0.86**	**515**

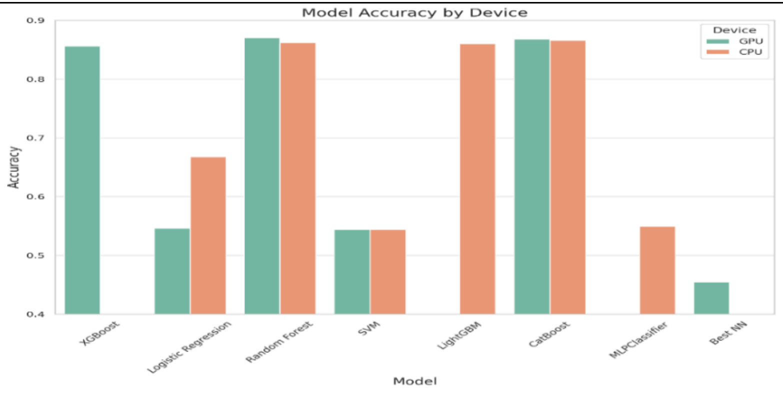
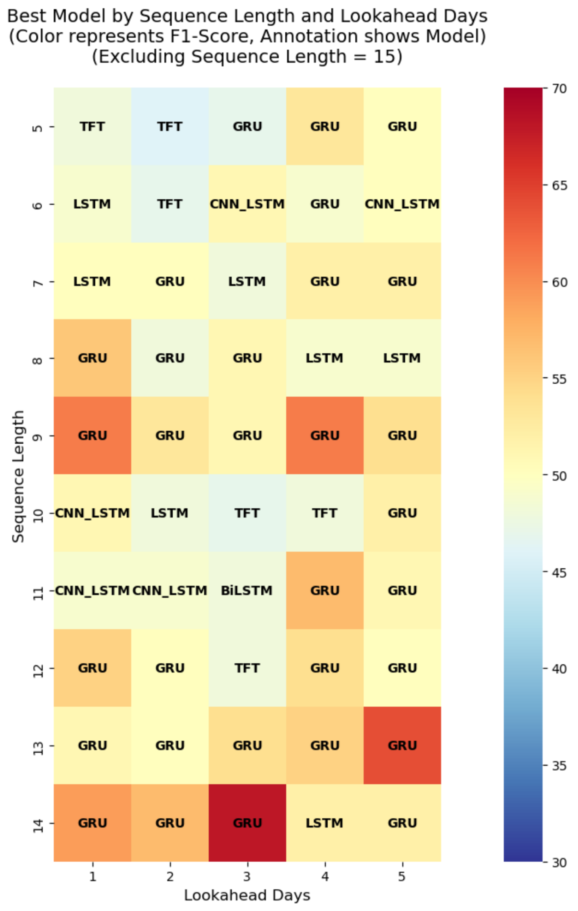


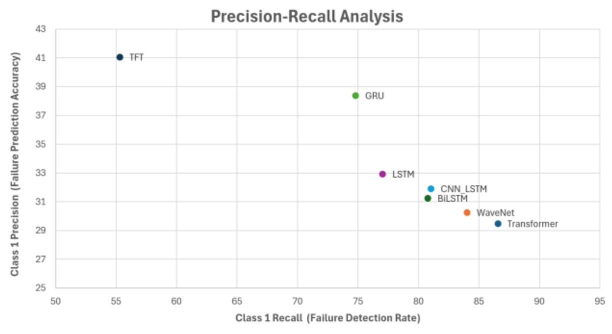
Fig. 3: Comparing Classification Models by Device

4.2. Time Series Models Classification Results

(a)



(b)



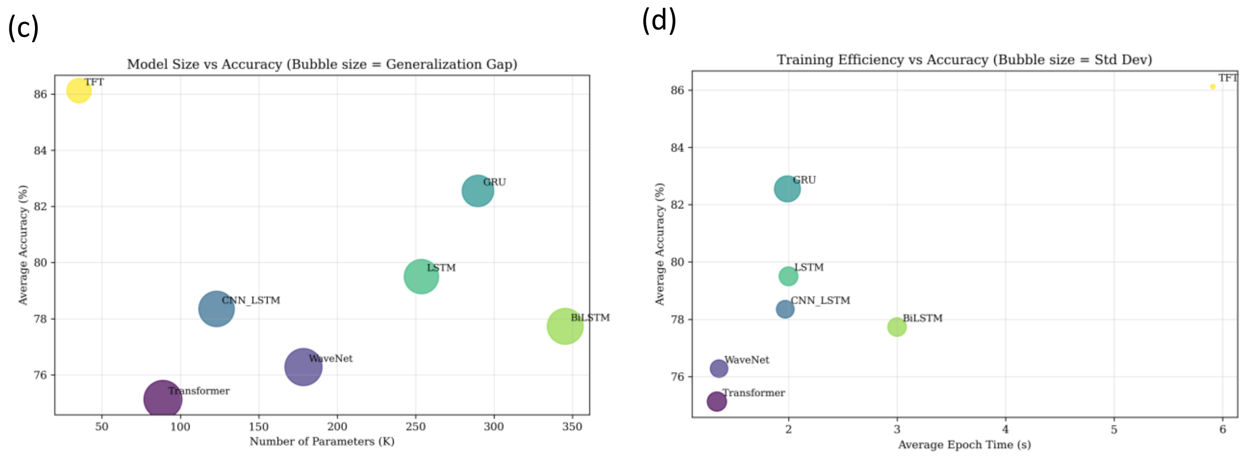


Fig. 4: Time Series Classification Models performance: (a) Temporal Analysis showing best model for each sequence length and look-ahead days for the F1-score for the Fail class, (b) Precision-Recall Analysis, (c) Model size vs Accuracy, and (c) Training Efficiency vs Accuracy

The third set of experiments compared seven deep learning architectures for time series classification models on different sequence lengths and look-ahead days. The different configurations created 350 experiments across sequence lengths of 5-14 days and lookahead periods of 1-5 days. The models evaluated as con **Fig.4d** in **Table 3** achieved overall test accuracy ranging from 60% to 92%, while F1-scores for failure detection varied between 31% and 68%, demonstrating significant performance variations across different architectural approaches.

The TFT model proved to be the most effective for predicting hard drive failures using S.M.A.R.T. data. Despite having the fewest parameters (35.4K), TFT had the highest computation overhead due to its architectural requirements. It achieved the highest average test accuracy (86.12%) and the smallest generalisation gap (10.28%), demonstrating superior robustness and efficiency. Its consistent performance across all configurations—coupled with the lowest standard deviation in accuracy (0.16)—indicates a strong ability to generalise without overfitting, making it particularly suitable for real-world deployment where model stability and parameter efficiency are critical.

The GRU model attained the highest fail-class F1-score (50.08%), indicating better sensitivity to impending failures. It exhibited higher variance in performance (StdDev accuracy: 5.58) and a larger generalisation gap (17.14%). More complex architectures, such as the Transformer and WaveNet, significantly underperformed, suffering from severe overfitting and instability, which suggests that their inductive biases are misaligned with the temporal structure and signal-to-noise ratio inherent in S.M.A.R.T. data.

The temporal parameter analysis revealed that sequence length and lookahead period showed varying optimal configurations across architectures. The best models for different metrics, varying sequence lengths, and look-ahead days are illustrated in **Fig. 4 (a)**. Increasing the sequence length had a limited impact overall; however, BiLSTM and WaveNet showed the most improvement. LSTM/GRU performed very low on shorter sequence lengths (5–7 days), then improved around 10–12 days; CNN-LSTM improves up to ~10 days, then levels off. While the Transformer performance is consistent across sequence lengths, validating its insensitivity to sequence window choice. As lookahead days increased (prediction horizon), performance degraded across all models, reinforcing that short-term failure forecasts (1–3 days) are more reliable than longer-term predictions in this dataset. However, Transformer remained stable, although not the top-performing model, and BiLSTM and WaveNet maintained the most resilience at 4–5 days, though with noticeable drops.

The precision-recall analysis revealed distinct trade-offs among models, with some architectures favouring high precision (reducing false alarms) while others prioritised recall (maximising failure detection). **Fig. 4 (b)** shows a Pareto frontier: as we move to the right to capture more failures (higher recall), we are forced to accept a higher rate of false alarms (lower precision). No model can dominate in both metrics simultaneously. The High-Precision / Low-Recall Cluster (Top-Left) includes TFT and GRU, which are ideal for extremely cost-sensitive operations where the expense of unnecessary drive replacements (labour, parts) is the primary concern. TFT would be chosen if operational efficiency is the absolute top priority. The Balanced Cluster (Middle) comprises LSTM, CNN-LSTM, and Bi-LSTM models. These are the best general-purpose choices for most predictive maintenance scenarios. It offers a good balance, reducing operational costs from false alarms while also providing a strong safety net by catching most failures. Finally, the High-Recall / Low-Precision Cluster (Bottom-Right) includes WaveNet and Transformer models. These are useful for safety-critical or high-availability systems where the cost of a missed failure and subsequent downtime is unacceptably high (e.g., core banking systems, medical data storage).

The computational efficiency analysis reveals a compelling inverse relationship between computational efficiency and generalisation performance, as illustrated in **Fig. 4 (c) and (d)**. The most parameter-heavy models like BiLSTM (345.4K params) and GRU (289.7K params) are computationally expensive per epoch yet exhibit some of the most significant generalisation gaps (22.2, 17.1) and high variance in performance (StdDev Accuracy: 5.58 for GRU), indicating poorer stability and overfitting. In stark contrast, the most efficient model, the Transformer (88.9K params), trains the fastest (1.34s/epoch) but achieves the lowest F1 score (44.0%) and the highest generalisation gap (24.8%), suggesting it may be underfitting the failure class. Striking an optimal balance is the TFT, which, despite having the fewest parameters (35.4K) and the slowest training time per epoch (5.91s), demonstrates superior overall performance. It achieves the highest test accuracy (86.1), the smallest generalisation gap (10.3), and the lowest variance (StdDev Accuracy: 0.16), proving to be the most robust, stable, and well-generalising model, likely due to its more complex internal architecture that requires longer to train but learns a more effective representation.

These results provide crucial insights to consider when implementing deep learning-based predictive maintenance systems in storage and computational infrastructure, sequence lengths for failure buildup analysis, and possible predictive horizon enabling informed decisions regarding model architecture selection, hyperparameter optimisation, and deployment considerations based on accuracy requirements, computational constraints, and operational priorities. TFT emerged as the best overall model for predictive maintenance on SMART data under constrained parameter budgets in the experiments presented in this paper. The detailed experimental results of the time series classification models are presented in Supplement 2.

4.3. Anomaly Detection Models Results

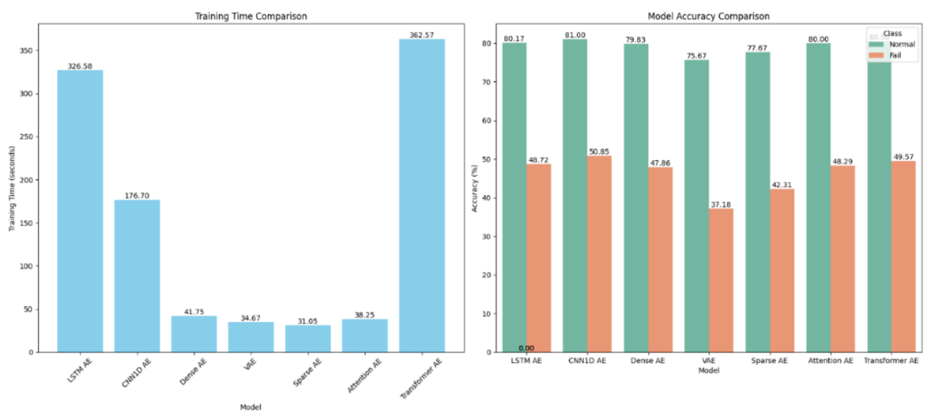


Fig. 5: AutoEncoder Models Comparison of Training Time and Accuracy Comparison for both Normal and Fail Classes

The performance of seven autoencoder (AE) models was compared in the fourth set of experiments. The autoencoder model's reconstruction error of each model was used to identify a threshold between the normal class and the anomalies / fail class. Each model suggested a threshold based on its reconstruction error and our prior knowledge of the fail class of this supervised dataset. **Fig.** 5 shows that the LSTM Autoencoder (LSTM AE), which took 326.58 seconds to train, achieved an accuracy of 80.17% for normal instances but only 48.72% for failure detection. This suggests that while the LSTM AE effectively captures sequential patterns, it struggles with detecting anomalies. The recommended threshold of 0.1448 reflects a balanced trade-off between sensitivity and specificity.

The CNN1D Autoencoder (CNN1D AE), trained in 176.70 seconds, had the highest accuracy for failure detection at 50.85% and slightly better accuracy for normal instances at 81%. Its performance suggests that convolutional layers are effective for capturing local patterns, with a shorter training time compared to LSTM AEs. However, its threshold of 0.1608 indicates a higher sensitivity to false positives. The Dense Autoencoder (Dense AE), which was trained in just 41.75 seconds, converged quickly but showed the lowest failure detection accuracy at 47.86%. While efficient, it may lack the capacity to model complex patterns required for effective anomaly detection. The threshold of 0.0743 reflects both its efficiency and its limitations in detecting rare events. The Variational Autoencoder (VAE), which required 34.67 seconds for training, achieved the lowest accuracy for both normal (75.67%) and failure classes (37.18%). Its probabilistic approach introduces variability that likely impacts its performance in distinguishing failures. The high threshold of 0.4093 suggests compromised reconstruction quality. The Sparse Autoencoder (Sparse AE), with a training time of 31.05 seconds, outperformed the VAE, achieving 77.67% accuracy for normal instances and 42.31% for failure detection. The sparsity regularisation aids feature extraction but may still fall short in capturing all necessary patterns. The threshold of 0.3387 represents a moderate balance between sensitivity and specificity. The Attention-Based Autoencoder (Attention AE), which was trained in 38.25 seconds, achieved similar accuracy rates to the LSTM AE for normal (80.00%) and failure (48.29%) classes. Integrating attention mechanisms enhances feature focus and overall performance, with a threshold of 0.0832 indicating a balanced sensitivity.

Finally, the Transformer Autoencoder (Transformer AE), requiring the longest training time of 362.57 seconds, achieved the highest failure detection accuracy of 49.57% and competitive performance for normal instances at 80.50%. The Transformer's self-attention mechanism effectively captures complex relationships, with a threshold of 0.0842 reflecting a nuanced balance between accuracy and training time. As seen with the Transformer AE, longer training times often correlate with higher accuracy. In comparison, models with shorter training times, like Sparse AE and Dense AE, offer quicker convergence but at the expense of lower performance. These results highlight the importance of striking a balance between model complexity, training efficiency, and accuracy, tailored to specific application requirements. The detailed experimental results of the anomaly detection models are presented in Supplement 2.

4.4. Hardware Considerations

Hardware constraints significantly influenced experimental design and dataset selection. Shared academic clusters impose memory and storage limitations, restricting large-scale experimentation. Google Colab's A100 GPU runtime (80 GB memory) proved suitable for processing annual data (four quarters), consuming over 500 compute units across initial machine learning tasks. Regression analyses using decision trees and random forests were conducted on a laptop equipped with an 11th Gen Intel Core i5-1135G7 processor and 16 GB of DDR4 memory. Classification tasks involving single-day SMART readings, 5-day sequences, and autoencoders (LSTM and 1D CNN) were utilised on NVIDIA's DLi Azure Cloud platform, which consisted of 48 Intel Xeon Platinum 8259CL CPUs and four Tesla T4 GPUs (each with 16 GB of memory). Additional validation experiments employed Google Colab's NVIDIA A100-SXM4-40GB (12 Xeon CPUs, 80 GB RAM) and an HP OMEN laptop with Intel Core i9-14900HX processor and NVIDIA GeForce RTX 4080. XGBoost performance across hardware configurations demonstrated the critical importance of accelerator selection and library optimisation **(Table 6 and Fig. 6).** GPU acceleration achieved speedups of 204.7%–5055.3%

over CPU implementations, although cuDF utilisation failed on the RTX 4080 due to memory constraints (143 MB available versus 1.24 GB required).

Table 6: XGBoost performance using different dataframes libraries on different platforms

Nvidia DLi MS Azure Platform	Google Colab	HP OMEN
CPU: 60.5 seconds	CPU: 67.8 seconds	CPU :18.0 seconds
GPU (no cuDF): 21.9 seconds	GPU (no cuDF) 5.1 seconds	GPU (no cuDF):8.8 seconds
GPU (cuDF) : 13.2 seconds	GPU (cuDF): 1.3 seconds	GPU (cuDF): Not Enough memory
Total speed-up with RAPIDS: 459.6%	Total speed-up with RAPIDS: 5055.3%	Total speed-up with RAPIDS: 204.7 %

To quantitatively evaluate GPU capabilities, we developed a weighted scoring model incorporating seven hardware characteristics:

$$\textit{Strength} = 0.25 \times \textit{Memory Bandwidth} + 0.20 \times \textit{Tensor Performance} + 0.15 \times \textit{VRAM Capacity} + 0.15 \times \textit{FP32 Performance} + 0.10 \times \textit{Power Efficiency} + 0.10 \times \textit{Multi-GPU Scaling} + 0.05 \times \textit{Architecture Features}$$

where components are normalised against maximum values. This framework prioritises memory bandwidth (favouring HBM2e architecture) and tensor performance while accounting for multi-GPU diminishing returns (80% efficiency per additional GPU). As **Table 7** demonstrates, Google Colab's A100 achieved superior performance (score: 80.5) through exceptional memory bandwidth (1555 GB/s) and tensor throughput (312 TFLOPS TF32), outperforming Azure's quadruple Tesla T4 configuration (78.4) despite lower FP32 throughput. The consumer-grade RTX 4080 (57.9) ranked third, constrained by memory bandwidth (716.8 GB/s) despite strong FP32 performance. These results establish that architectural advantages in memory systems and specialised AI accelerators outweigh theoretical peak performance for data-intensive applications, providing a validated hardware selection framework for predictive maintenance implementations.

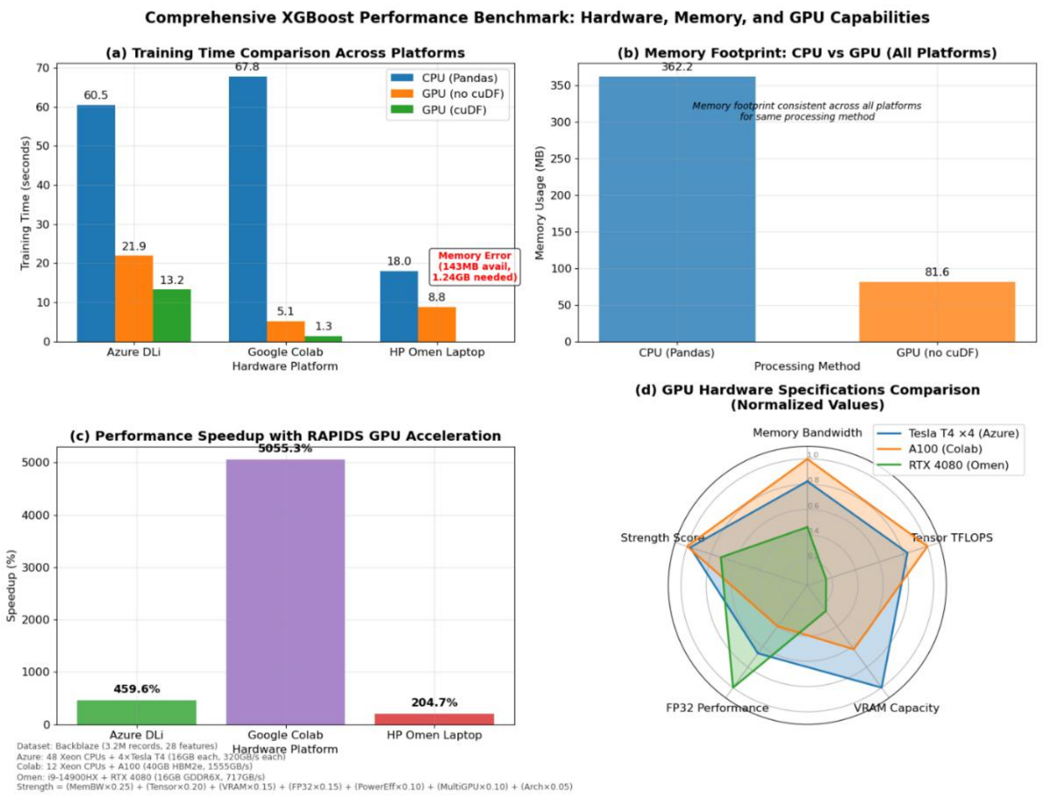


Fig. 6: Comparing XGBoost Classification using different Dataframe structures in both CPU and GPU of the various platforms and hardware configurations

Table 7: GPU Hardware Specifications

GPU Model	Memory (GB)	Memory Type	Memory Bandwidth (GB/s)	FP32 TFLOPS	Tensor TFLOPS (FP16)	CUDA Cores	GPU Count	TDP (W)	Strength	Architecture
Tesla T4 ×4 (Azure)	64	GDDR6	1280 (4×320)	32.4 (4×8.1)	260 (4×65)	10240 (4×2560)	4	280	78.4	Turing
A100 (Colab)	40	HBM2e	1555	19.5	312 (TF32)	6912	1	250	**80.5**	Ampere
RTX 4080 (Omen)	16	GDDR6X	716.8	48.74	48.74 (FP16)	9728	1	320	57.9	Ada Lovelace

5. DISCUSSIONS AND FUTURE RESEARCH DIRECTIONS

This comprehensive study presents the most extensive comparative analysis of machine learning models for predictive maintenance applications to date, systematically evaluating traditional algorithms, deep learning architectures, and autoencoder-based approaches across four distinct predictive maintenance tasks. Our findings provide crucial insights into model selection criteria, computational trade-offs, and practical implementation considerations that will guide future research and industrial deployment decisions.

5.1 Key Findings and Contributions

This study presents a comprehensive benchmark of machine learning approaches for HDD failure prediction, evaluating over 20 models across four distinct tasks: regression, classical classification, deep time-series classification, and anomaly detection (AE). The consolidated results reveal critical insights into model efficacy, efficiency, and suitability for real-world deployment.

In the regression task, tree-based ensembles demonstrated superior performance, with a fine-tuned Random Forest model achieving the lowest MSE (35.98), significantly outperforming linear models and deep neural networks, which suffered from severe overfitting. For classical classification on a highly imbalanced dataset, tree-based methods (Random Forest: 86.99%, XGBoost: 85.63%) again proved most effective and efficient, particularly when accelerated with RAPIDS on GPU, achieving a 50x speed-up. However, their high accuracy was misleading, as the failure class F1-score remained low (~0.05), highlighting the inherent challenge of extreme class imbalance and the inadequacy of accuracy as a sole metric.

The most significant findings emerged from the deep learning experiments. The time-series classification task definitively established the Temporal Fusion Transformer (TFT) as the optimal architecture, achieving the highest average accuracy (86.12%) and smallest generalisation gap (10.28%) with the fewest parameters (35.4K), underscoring its robustness and efficiency. The Gated Recurrent Unit (GRU) performed best in terms of failure detection sensitivity, achieving the highest fail-class F1-score (50.08) and a peak accuracy of 92.52%. Conversely, the anomaly detection study found that all autoencoder variants, including a Transformer-based AE, delivered only moderate performance (Test Accuracy: 65-73%), failing to surpass the supervised deep learning models and reinforcing that supervised learning on temporal sequences is the most promising path for accurate HDD prognostics.

In conclusion, the key contribution of this paper is a definitive model ranking that guides practitioners. For tabular SMART data, tree-based models are a strong baseline. For leveraging temporal sequences, TFT is the most robust and efficient choice, while GRU should be considered for maximising failure recall. The poor performance of AEs and other complex architectures, such as vanilla Transformers, suggests that architectural simplicity aligned with the data structure is paramount for this domain.

5.2 Practical Implications for Industrial Implementation

Our findings provide actionable guidance for organisations implementing machine learning-based predictive maintenance systems, with clear recommendations for different deployment scenarios and constraints.

Resource-Constrained Environments: For organisations with limited computational resources or edge computing requirements, our results strongly support the adoption of ensemble methods as first-choice algorithms. Random Forest and XGBoost provide excellent performance with reasonable computational overhead and minimal hyperparameter sensitivity, making them suitable for deployment in resource-constrained industrial environments.

The competitive performance of CPU-based implementations for these algorithms ensures that effective predictive maintenance systems can be deployed without requiring expensive GPU hardware, democratizing access to advanced predictive maintenance capabilities across different organisational scales and budgets.

High-Performance Applications: For applications where predictive accuracy is paramount and computational resources are abundant, our results support the adoption of CNN-LSTM architectures for temporal sequence analysis, combined with ensemble methods for immediate failure detection and RUL prediction. This hybrid approach leverages the strengths of different algorithmic paradigms to achieve optimal performance across multiple predictive maintenance tasks.

Scalability and Maintenance Considerations: The favourable scaling properties of ensemble methods and the efficiency of modern deep learning frameworks suggest that the approaches evaluated in this study can be successfully scaled to larger industrial datasets and more complex monitoring systems. However, the importance of ongoing hyperparameter optimisation and model maintenance highlighted by our results necessitates systematic approaches to model lifecycle management in industrial deployments.

5.3 Limitations and Methodological Considerations

While this study provides an extensive evaluation of ML and DL models using the Backblaze dataset, several limitations must be acknowledged to contextualise the findings and guide future research. The dataset, although large and real-world, is domain-specific, making it challenging to drive failures in data centres. This restricts the generalizability of our results to other industrial contexts, such as manufacturing, energy, or transportation, where equipment heterogeneity, multimodal sensor data (e.g., vibration, thermal, and acoustics), and diverse failure mechanisms (e.g., corrosion, fatigue cracks, and electrical faults) are prevalent. The absence of such variability limits the assessment of model robustness under real-world conditions, which involve data volume, velocity, variety, and veracity—key challenges in industrial IoT deployments.

Temporal Scope Limitations: Our analysis focuses on relatively short temporal sequences (5-12 days) and prediction horizons (1-5 days), which align with the characteristics of hard drive failures; however, this may not be optimal for equipment with longer degradation cycles. Industrial equipment such as large turbines, generators, or process vessels (large containers used in industrial production, like chemical reactors or storage tanks) may exhibit degradation patterns that develop over months or years, requiring different temporal modelling approaches.

Class Imbalance Treatment: While our down-sampling approach proved effective for creating balanced training sets, this strategy may not be optimal for all industrial applications. The choice of balancing strategy can significantly impact model performance and should be carefully considered based on the specific costs of false positives versus false negatives in each application domain.

5.4 Future Research Directions

Our comprehensive evaluation reveals several promising directions for future research that could significantly advance the field of machine learning-based predictive maintenance.

5.4.1 Advanced Architectural Innovations

Hybrid Architecture Development: The superior performance of CNN-LSTM architectures suggests significant potential for developing more sophisticated hybrid approaches that combine the strengths of different deep learning paradigms. Future research should explore architectures that integrate attention mechanisms with convolutional and recurrent components, potentially achieving better performance than any single approach.

Graph neural networks represent another promising direction for modelling complex relationships between multiple sensors and equipment components. Industrial systems often exhibit complex interdependencies that could be effectively captured through graph-based representations, enabling more comprehensive failure prediction across interconnected equipment networks [49].

Physics-Informed Machine Learning: The integration of domain knowledge and physical principles with data-driven approaches represents an auspicious research direction. Physics-informed neural networks that incorporate thermodynamic principles, mechanical stress models, or electrical system behaviours could achieve better generalisation and require less training data than purely data-driven approaches. Such approaches could be particularly valuable for equipment types where physical degradation models are well-understood, enabling the combination of mechanistic understanding with machine learning flexibility to achieve superior predictive performance [50].

Recent studies focus on **Deep Q-Networks (DQN) and Proximal Policy Optimisation (PPO)** for dynamic maintenance decisions. These algorithms optimise maintenance schedules by learning from real-time sensor data, adapting to changing conditions like equipment degradation or operational demands [51]. Hyperparameter tuning via Bayesian methods (e.g., Optuna) significantly enhances PPO's performance, reducing energy consumption and downtime in metro systems [52]. Algorithms like Sequential Multi-Objective Multi-Agent PPO (SMOMA-PPO) optimise trade-offs between remaining useful life (RUL) and inspection costs, achieving ~15% reduction in RUL waste and ~10% longer inspection intervals [53]. Combining RL (Reinforcement Learning) with other ML techniques enhances robustness, such as DRL (Deep RL) + Random Forest + Gradient Boosting.

5.4.2 Multi-Modal and Multi-Scale Analysis

Sensor Fusion and Multi-Modal Learning: Future research should explore approaches for effectively combining different types of sensor data, including vibration, temperature, acoustic, and visual monitoring systems. Multi-modal learning approaches that can jointly process heterogeneous sensor data types could provide a more comprehensive equipment health assessment than single-modality approaches [54].

The development of **attention mechanisms** specifically designed for multi-modal sensor fusion could enable models to dynamically focus on the most informative sensor types for different failure modes, thereby improving both accuracy and interpretability [55].

Multi-Scale Temporal Modelling: Our findings regarding the importance of temporal parameter optimisation suggest significant potential for developing adaptive approaches that can automatically determine optimal sequence lengths and prediction horizons for different equipment types and failure modes [56].

Hierarchical temporal modelling approaches that simultaneously capture short-term fluctuations, medium-term trends, and long-term degradation patterns can provide a more comprehensive understanding of temporal dynamics than fixed-scale approaches.

5.4.3 Explainable AI and Interpretability

Model Interpretability Enhancement: The practical deployment of machine learning-based predictive maintenance systems requires models that can provide explanations for their predictions, particularly in safety-critical applications. Future research should focus on developing interpretability techniques specifically designed for predictive maintenance applications.

Attention visualisation techniques for deep learning models could provide insights into which sensors and time periods are most indicative of impending failures, enabling maintenance personnel to better understand and trust model predictions. Similarly, feature importance analysis for ensemble methods could identify the most critical monitoring parameters for different failure modes.

Uncertainty Quantification: The development of approaches for quantifying prediction uncertainty represents a critical need for the practical deployment of predictive maintenance. Bayesian deep learning approaches or ensemble-based uncertainty estimation could provide confidence intervals for failure predictions, enabling more informed maintenance decision-making [57].

5.4.4 Adaptive and Continual Learning

Online Learning and Model Adaptation: The characteristics and operating conditions of industrial equipment evolve over time due to ageing, maintenance activities, and changing operational requirements. Future research should explore online learning approaches that can continuously adapt predictive maintenance models based on new data and feedback from maintenance activities. Continual learning techniques that can incorporate new failure modes without forgetting previously learned patterns could enable predictive maintenance systems to evolve and improve over their operational lifetime.

Transfer Learning and Domain Adaptation: The development of approaches for transferring knowledge between different equipment types, operating conditions, or industrial environments could significantly reduce the data requirements for deploying predictive maintenance systems in new contexts.

Meta-learning approaches focus on developing models that can quickly adapt to new tasks with minimal data, making it particularly useful when introducing new equipment. These approaches could democratize access to advanced predictive maintenance capabilities across diverse industrial applications [58]. Adversarial Learning enhances models' robustness against noisy or adversarial sensor data, ensuring reliability even under suboptimal conditions. Training models with adversarial techniques can maintain accuracy in industrial settings where data quality may be compromised [59].

5.4.5 Integration with Industrial IoT and Edge Computing

Edge Computing Optimisation: The increasing deployment of edge computing in industrial environments necessitates research into model compression and optimisation techniques specifically designed for predictive maintenance applications. Techniques such as knowledge distillation, pruning, and quantisation could enable the deployment of sophisticated models on resource-constrained edge devices.

Federated Learning for Predictive Maintenance: The development of federated learning approaches that can train predictive maintenance models across multiple industrial sites while preserving data privacy represents an important research direction. Such approaches could enable the development of collaborative models that benefit from diverse operational data while addressing industrial data security concerns [60].

5.5 Concluding Remarks

This study establishes a rigorous, multi-faceted benchmark for machine learning in predictive maintenance, providing empirical evidence to guide model selection. The key conclusion is that optimal architecture is highly dependent on data structure and operational objectives: tree-based ensembles excel on static tabular data, while deep temporal models are paramount for sequential sensor data.

Our findings specifically demonstrate the superiority of the Temporal Fusion Transformer (TFT) for robust multi-horizon forecasting and the Gated Recurrent Unit (GRU) for maximising failure recall. These results, coupled with the quantified computational trade-offs between CPU and GPU implementations, offer a clear framework for developing efficient and reliable prognostic systems.

Future work should focus on integrating these models into scalable, real-time monitoring frameworks and exploring their generalizability across other industrial assets and failure modes. By advancing these research directions, this work contributes a foundational step towards more resilient, data-driven industrial operations, ultimately enhancing reliability and sustainability.

FUNDING INFORMATION

This research received no specific grant from any funding agency.

CONFLICT OF INTEREST

The authors have no financial interest to declare in relation to the content of this article.

DATA AVAILABILITY

The data used and/or analyzed during the current study is available from the author on reasonable request

REFERENCES

[1] T. P. Carvalho, F. A. A. M. N. Soares, R. Vita, R. D. P. Francisco, J. P. Basto, and S. G. S. Alcalá, ‘A systematic literature review of machine learning methods applied to predictive maintenance’, *Computers & Industrial Engineering*, vol. 137, p. 106024, Nov. 2019, doi: 10.1016/j.cie.2019.106024.

[2] T. Zhu, Y. Ran, X. Zhou, and Y. Wen, ‘A Survey of Predictive Maintenance: Systems, Purposes and Approaches’, *IEEE Communications Surveys & Tutorials*, vol. 20, pp. 1–36, Dec. 2019.

[3] Deloitte, ‘Predictive Maintenance: Taking pro-active measures based on advanced data analytics to predict and avoid machine failure’, Deloitte Analytics Institute, Oct. 2022. Accessed: Aug. 15, 2024.

[4] UptimeAI, ‘Predictive Maintenance Techniques’, UptimeAI. Accessed: Aug. 15, 2024.

[5] B. Wright, ‘AI Is Here, and It’s Helping With Predictive Maintenance in the Oil Field’, JPT. Accessed: Aug. 26, 2025.

[6] M. dos S. Póvoas *et al.*, ‘Artificial Intelligence in the Oil and Gas Industry: Applications, Challenges, and Future Directions’, *Applied Sciences*, vol. 15, no. 14, p. 7918, Jan. 2025, doi: 10.3390/app15147918.

[7] M. Wang, X. Su, H. Song, Y. Wang, and X. Yang, ‘Enhancing predictive maintenance strategies for oil and gas equipment through ensemble learning modeling’, *J Petrol Explor Prod Technol*, vol. 15, no. 3, p. 46, Feb. 2025, doi: 10.1007/s13202-025-01931-x.

[8] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*, Fifth edition. in Wiley series in probability and statistics. Hoboken, New Jersey: John Wiley & Sons, Inc, 2016.

[9] C. C. Holt, ‘Forecasting seasonals and trends by exponentially weighted moving averages’, *International Journal of Forecasting*, vol. 20, no. 1, pp. 5–10, Jan. 2004, doi: 10.1016/j.ijforecast.2003.09.015.

[10] P. R. Winters, ‘Forecasting Sales by Exponentially Weighted Moving Averages’, *Management Science*, vol. 6, no. 3, pp. 324–342, Apr. 1960, doi: 10.1287/mnsc.6.3.324.

[11] A. C. Harvey, *Forecasting, Structural Time Series Models and the Kalman Filter*, 1st edn. Cambridge University Press, 1990. doi: 10.1017/CBO9781107049994.

[12] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2005. doi: 10.7551/mitpress/3206.001.0001.

[13] E. A. Nadaraya, ‘On Estimating Regression’, *Theory Probab. Appl.*, vol. 9, no. 1, pp. 141–142, Jan. 1964, doi: 10.1137/1109020.

[14] A. J. Smola and B. Schölkopf, ‘A tutorial on support vector regression’, *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, Aug. 2004, doi: 10.1023/B:STCO.0000035301.49549.88.

[15] L. Breiman, ‘Random Forests’, *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[16] T. Chen and C. Guestrin, ‘XGBoost: A Scalable Tree Boosting System’, in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco California USA: ACM, Aug. 2016, pp. 785–794. doi: 10.1145/2939672.2939785.

[17] G. A. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi, ‘Machine Learning for Predictive Maintenance: A Multiple Classifier Approach’, *IEEE Trans. Ind. Inf.*, vol. 11, no. 3, pp. 812–820, June 2015, doi: 10.1109/TII.2014.2349359.

[18] S. Hochreiter and J. Schmidhuber, ‘Long Short-Term Memory’, *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[19] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, ‘LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection’, 2016, *arXiv*. doi: 10.48550/ARXIV.1607.00148.

[20] Z. Zainuddin, E. A. P. Akhir, and M. H. Hasan, ‘Predicting machine failure using recurrent neural network-gated recurrent unit (RNN-GRU) through time series data’, *Bulletin EEI*, vol. 10, no. 2, pp. 870–878, Apr. 2021, doi: 10.11591/eei.v10i2.2036.

[21] M. Schuster and K. K. Paliwal, ‘Bidirectional recurrent neural networks’, *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997, doi: 10.1109/78.650093.

[22] J. Rala Cordeiro, A. Raimundo, O. Postolache, and P. Sebastião, ‘Neural Architecture Search for 1D CNNs—Different Approaches Tests and Measurements’, *Sensors*, vol. 21, no. 23, p. 7990, Nov. 2021, doi: 10.3390/s21237990.

[23] G. Muthukumar and P. Jyosna, ‘CNN-LSTM Hybrid Deep Learning Model for Remaining Useful Life Estimation’, 2024, doi: 10.48550/ARXIV.2412.15998.

[24] A. Vaswani *et al.*, ‘Attention Is All You Need’, in *NeurIPS Proceedings*, LONG BEACH CA, Dec. 2017. doi: https://dl.acm.org/doi/10.5555/3295222.3295349.

[25] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, ‘A Time Series is Worth 64 Words: Long-term Forecasting with Transformers’, 2022, *arXiv*. doi: 10.48550/ARXIV.2211.14730.

[26] W. Luo, L. Yan, Y. Zhu, J. Ye, W. Pan, and X. Zou, ‘Transformer-based long-distance PDM optical fiber channel modeling method’, in *Second International Conference on Optical Communication and Optical Information Processing (OCOIP 2024)*, Y. Yue, Ed., Wuhan, China: SPIE, Mar. 2025, p. 17. doi: 10.1117/12.3060644.

[27] A. Borovykh, S. Bohte, and C. W. Oosterlee, ‘Conditional Time Series Forecasting with Convolutional Neural Networks’, Sept. 17, 2018, *arXiv*: arXiv:1703.04691. doi: 10.48550/arXiv.1703.04691.

[28] D. P. Kingma and M. Welling, ‘Auto-Encoding Variational Bayes’, Dec. 10, 2022, *arXiv*: arXiv:1312.6114. doi: 10.48550/arXiv.1312.6114.

[29] A. Ng, ‘Sparse autoencoder’, in *CS294A "Deep Learning and Unsupervised Feature Learning " Lecture notes*, vol. Winter 2011, 2011. Accessed: June 29, 2025. [Online]. Available: https://web.stanford.edu/class/cs294a/sparseAutoencoder_2011new.pdf

[30] D. Bahdanau, K. Cho, and Y. Bengio, ‘Neural Machine Translation by Jointly Learning to Align and Translate’, 2014, *arXiv*. doi: 10.48550/ARXIV.1409.0473.

[31] H. D. Nguyen, K. P. Tran, S. Thomassey, and M. Hamad, ‘Forecasting and Anomaly Detection approaches using LSTM and LSTM Autoencoder techniques with the applications in supply chain management’, *International Journal of Information Management*, vol. 57, p. 102282, Apr. 2021, doi: 10.1016/j.ijinfomgt.2020.102282.

[32] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, 'SMOTE: Synthetic Minority Over-sampling Technique', *jair*, vol. 16, pp. 321–357, June 2002, doi: 10.1613/jair.953.

[33] Haibo He and E. A. Garcia, 'Learning from Imbalanced Data', *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sept. 2009, doi: 10.1109/TKDE.2008.239.

[34] M. T. Ribeiro, S. Singh, and C. Guestrin, '"Why Should I Trust You?": Explaining the Predictions of Any Classifier', in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco California USA: ACM, Aug. 2016, pp. 1135–1144. doi: 10.1145/2939672.2939778.

[35] A. G. Howard *et al.*, 'MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications', 2017, *arXiv*. doi: 10.48550/ARXIV.1704.04861.

[36] Q. Yang, Y. Liu, T. Chen, and Y. Tong, 'Federated Machine Learning: Concept and Applications', *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, Mar. 2019, doi: 10.1145/3298981.

[37] M. Latifi, F. G. Darvishvand, O. Khandel, and M. L. Nowsoud, 'A deep reinforcement learning model for predictive maintenance planning of road assets: Integrating LCA and LCCA', 2021, *arXiv*. doi: 10.48550/ARXIV.2112.12589.

[38] M. Raissi, P. Perdikaris, and G. E. Karniadakis, 'Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations', *Journal of Computational Physics*, vol. 378, pp. 686–707, Feb. 2019, doi: 10.1016/j.jcp.2018.10.045.

[39] R. Van Dinter, B. Tekinerdogan, and C. Catal, 'Predictive maintenance using digital twins: A systematic literature review', *Information and Software Technology*, vol. 151, p. 107008, Nov. 2022, doi: 10.1016/j.infsof.2022.107008.

[40] P. O'Donovan, K. Leahy, K. Bruton, and D. T. J. O'Sullivan, 'An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities', *Journal of Big Data*, vol. 2, no. 1, p. 25, Dec. 2015, doi: 10.1186/s40537-015-0034-z.

[41] Madhukar Dharavath, 'AI-Driven Predictive Maintenance in Data Infrastructure: A Multi-Modal Framework for Enhanced System Reliability', *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol*, vol. 10, no. 6, pp. 824–834, Nov. 2024, doi: 10.32628/CSEIT241061118.

[42] A. Siddique, R. K. Kundu, G. R. Mode, and K. A. Hoque, 'RobustPdM: Designing Robust Predictive Maintenance against Adversarial Attacks', Aug. 10, 2023, *arXiv*: arXiv:2301.10822. doi: 10.48550/arXiv.2301.10822.

[43] B. Lindemann, T. Müller, H. Vietz, N. Jazdi, and M. Weyrich, 'A survey on long short-term memory networks for time series prediction', *Procedia CIRP*, vol. 99, pp. 650–655, 2021, doi: 10.1016/j.procir.2021.03.088.

[44] W. Song, C. Gao, Y. Zhao, and Y. Zhao, 'A Time Series Data Filling Method Based on LSTM—Taking the Stem Moisture as an Example', *Sensors*, vol. 20, no. 18, p. 5045, Sept. 2020, doi: 10.3390/s20185045.

[45] K. Ishida, A. Ercan, T. Nagasato, M. Kiyama, and M. Amagasaki, 'Use of 1D-CNN for input data size reduction of LSTM in Hourly Rainfall-Runoff modeling', Nov. 07, 2021, *arXiv*: arXiv:2111.04732. Accessed: May 02, 2024. [Online]. Available: http://arxiv.org/abs/2111.04732

[46] L. Lin, Z. Li, R. Li, X. Li, and J. Gao, 'Diffusion Models for Time Series Applications: A Survey', Apr. 30, 2023, *arXiv*: arXiv:2305.00624. Accessed: Aug. 15, 2024. [Online]. Available: http://arxiv.org/abs/2305.00624

[47] A. van den Oord *et al.*, 'WaveNet: A Generative Model for Raw Audio', 2016, *arXiv*. doi: 10.48550/ARXIV.1609.03499.

[48] B. Lim, S. O. Arik, N. Loeff, and T. Pfister, 'Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting', Sept. 27, 2020, *arXiv*: arXiv:1912.09363. doi: 10.48550/arXiv.1912.09363.

[49] L. Xia, P. Zheng, X. Li, Robert. X. Gao, and L. Wang, 'Toward cognitive predictive maintenance: A survey of graph-based approaches', *Journal of Manufacturing Systems*, vol. 64, pp. 107–120, July 2022, doi: 10.1016/j.jmsy.2022.06.002.

[50] L. G. Huber, T. Palmé, and M. A. Chao, 'Physics-Informed Machine Learning for Predictive Maintenance: Applied Use-Cases', in *2023 10th IEEE Swiss Conference on Data Science (SDS)*, Zurich, Switzerland: IEEE, June 2023, pp. 66–72. doi: 10.1109/SDS57534.2023.00016.

[51] K. Varalakshmi and J. Kumar, 'Optimized predictive maintenance for streaming data in industrial IoT networks using deep reinforcement learning and ensemble techniques', *Sci Rep*, vol. 15, no. 1, p. 27201, July 2025, doi: 10.1038/s41598-025-10268-8.

[52] M. H. Rziki, A. E. Hadbi, M. K. Boutahir, and M. C. Abounaima, ‘Adaptive Predictive Maintenance and Energy Optimization in Metro Systems Using Deep Reinforcement Learning’, *Sustainability*, vol. 17, no. 11, p. 5096, June 2025, doi: 10.3390/su17115096.

[53] Y. Chen and C. Liu, ‘Sequential Multi-objective Multi-agent Reinforcement Learning Approach for Predictive Maintenance’, Feb. 04, 2025, *arXiv*: arXiv:2502.02071. doi: 10.48550/arXiv.2502.02071.

[54] J. J. Montero Jimenez, S. Schwartz, R. Vingerhoeds, B. Grabot, and M. Salaün, ‘Towards multi-model approaches to predictive maintenance: A systematic literature survey on diagnostics and prognostics’, *Journal of Manufacturing Systems*, vol. 56, pp. 539–557, July 2020, doi: 10.1016/j.jmsy.2020.07.008.

[55] R. De Luca, A. Ferraro, A. Galli, M. Gallo, V. Moscato, and G. Sperlì, ‘A deep attention based approach for predictive maintenance applications in IoT scenarios’, *JMTM*, vol. 34, no. 4, pp. 535–556, May 2023, doi: 10.1108/JMTM-02-2022-0093.

[56] Y. Fan, Q. Tang, Y. Guo, and Y. Wei, ‘BiLSTM-MLAM: A Multi-Scale Time Series Prediction Model for Sensor Data Based on Bi-LSTM and Local Attention Mechanisms’, *Sensors*, vol. 24, no. 12, p. 3962, June 2024, doi: 10.3390/s24123962.

[57] J. U. Umavezi, ‘Bayesian Deep Learning for Uncertainty Quantification in Financial Stress Testing and Risk Forecasting’, *Int. J. Res. Publ. Rev.*, vol. 6, no. 5, pp. 6540–6555, May 2025, doi: 10.55248/gengpi.6.0525.1786.

[58] B. Deprez, ‘Meta learning for predictive maintenance under varying operating conditions’, Master’s dissertation submitted in order to obtain the academic degree of Master of Science in Information Engineering Technology, GHENT University, Belgium, 2021. Accessed: Aug. 16, 2024.

[59] C. Liu, D. Tang, H. Zhu, and Q. Nie, ‘A Novel Predictive Maintenance Method Based on Deep Adversarial Learning in the Intelligent Manufacturing System’, *IEEE Access*, vol. 9, pp. 49557–49575, 2021, doi: 10.1109/ACCESS.2021.3069256.

[60] S. Saha, A. Hota, A. K. Chattopadhyay, A. Nag, and S. Nandi, ‘A multifaceted survey on privacy preservation of federated learning: progress, challenges, and opportunities’, *Artif Intell Rev*, vol. 57, no. 7, p. 184, June 2024, doi: 10.1007/s10462-024-10766-7.